GPM

GLOBAL PRECIPITATION MEASUREMENT

Global Precipitation Measurement IV&V

GPM Health & Safety (HS) Dynamic Software Testing

September 15, 2011

Phil Wheeler
philip.n.wheeler@ivv.nasa.gov
304-472-9520 x1211

Greg Black
gregory.j.black@ivv.nasa.gov
681-753-5253

IV&V PM: Eric Sylvania
IV&V Lead Analyst: Amy Robinson

Developer: GSFC/JAXA
Launch Date: July 2013

Agenda

- **GPM Mission Overview**
- **Testing in ITC**
- **GPM Health & Safety (HS) Application Context**
- **Understanding the HS Application**
- **Formulating New HS Tests to Run in ITC**
- **Results Presented to the Project**
  - **Example 1:** Wakeup Message Pipe
  - **Example 2:** cFE power-on reset processing
- **Lessons Learned**
- **Process Forward** – Improvements to GPM IV&V Dynamic Testing
- **Conclusion**

Mission Overview

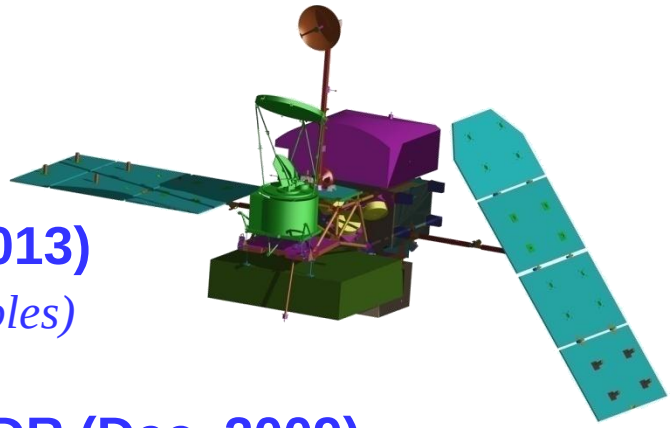
Mission Objectives:

- Advance precipitation measurement capability from space
- Improve knowledge of precipitation systems, water cycle variability, and fresh water availability
- Improve climate and hydrometeorological modeling, prediction, and 4-D climate reanalysis

Instruments:

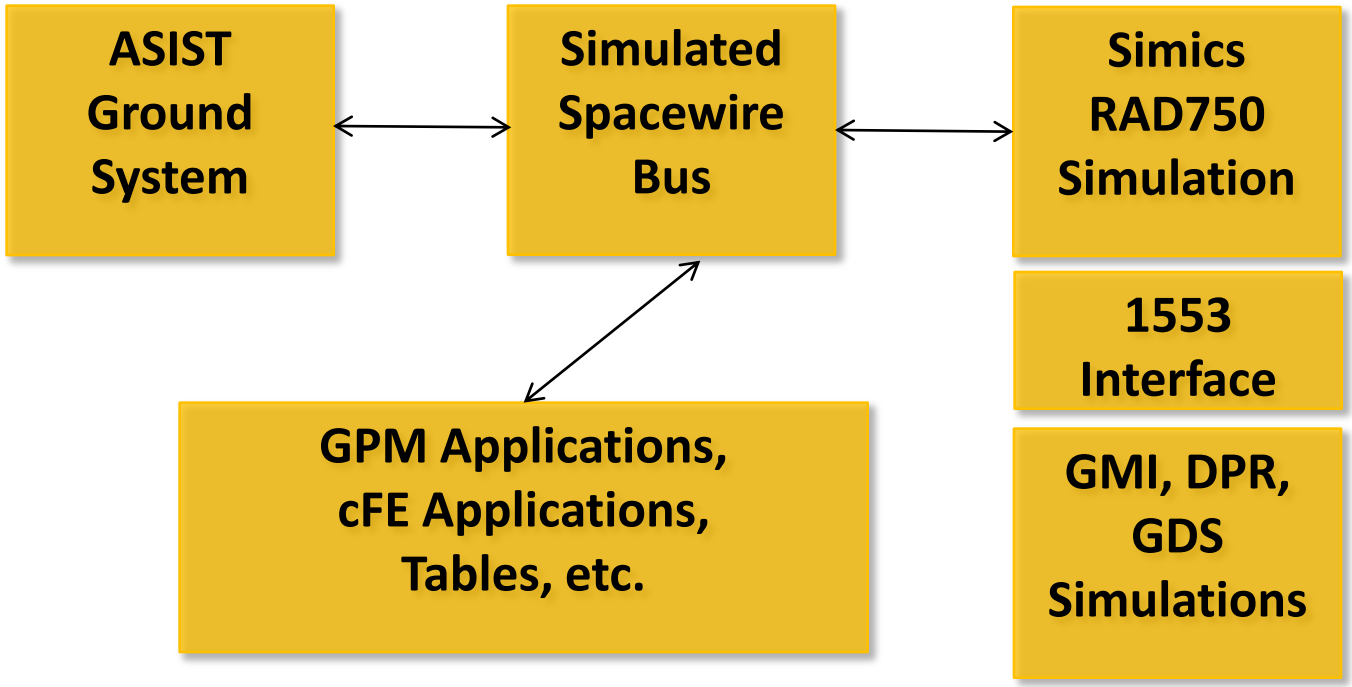
- Dual-Frequency Precipitation Radar (DPR)
- GPM Microwave Imager (GMI)

NASA Center: **GSFC**
Website: gpm.gsfc.nasa.gov
Major Developers: **JAXA; Ball; GSFC**
Launch: **Tanegashima (July, 2013)**
Mission Duration: **3 years** *(5 years consumables)*
Mission Phase: **Implementation**
Mission Status: **Completed Mission CDR (Dec, 2009)**

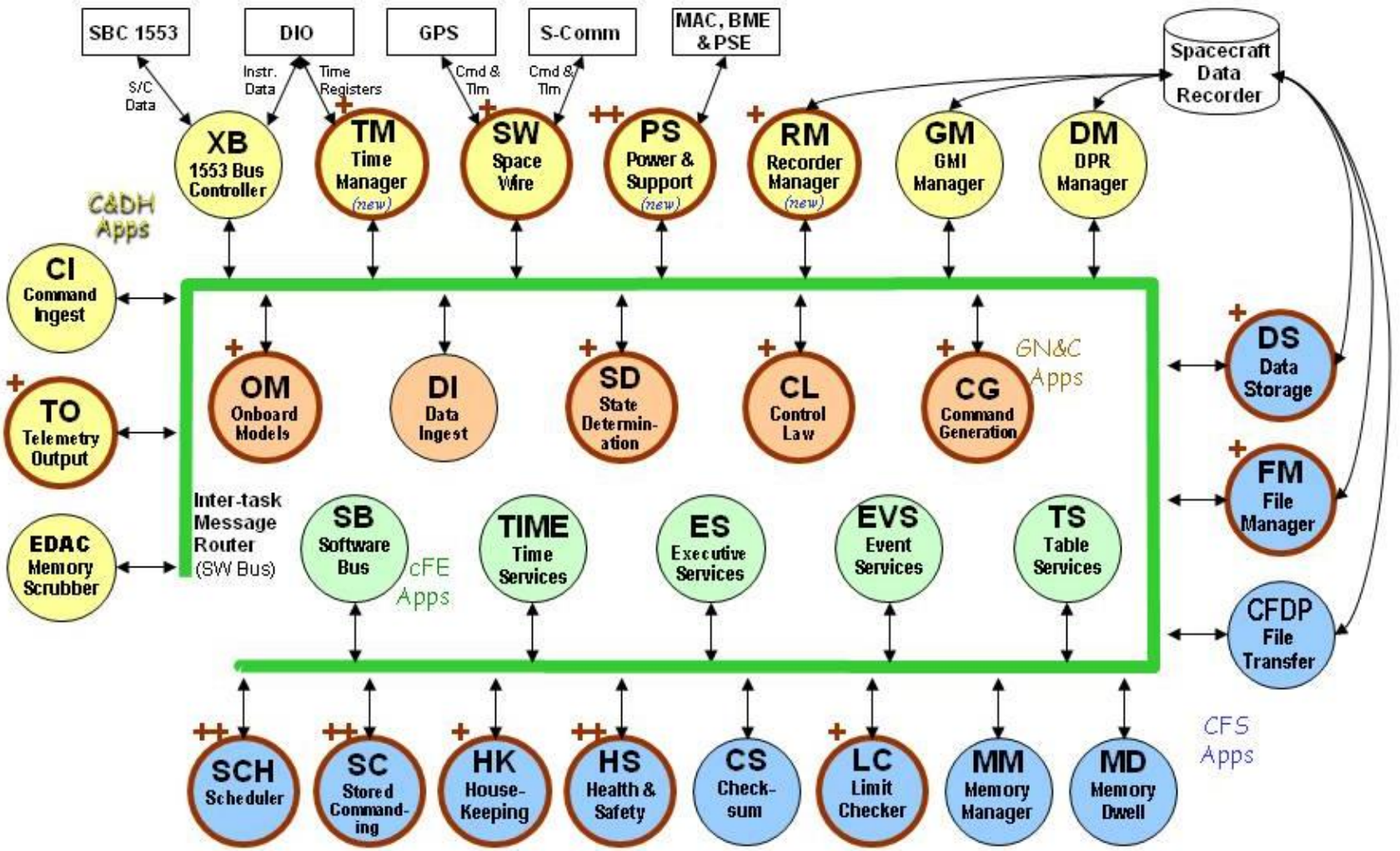


ITC Testing

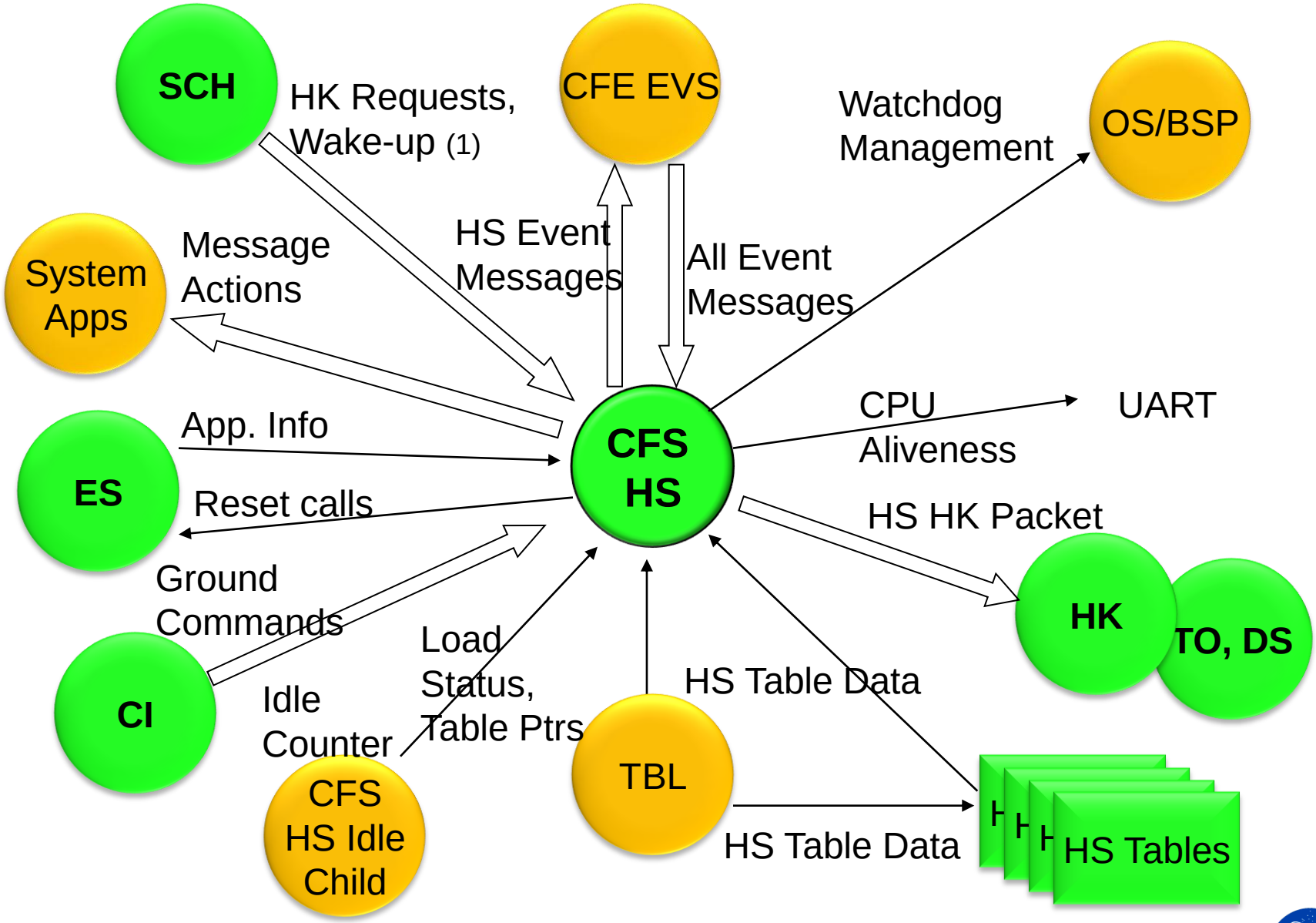
- The Independent Test Capability (ITC) lab has been developed as a new tool available to IV&V teams in order to validate and verify FSW in a more robust way.
- The GPM IV&V project is the pilot project for ITC testing.



Software Context Diagram



HS Application Context within GPM



(1) Wake-up event is not a GPM requirement

Understanding the HS Application

- **HS is an application intended to be reused on future missions**
  - HS files customized for GPM are separated into a “build” directory
- **HS is a critical application process that can trigger a processor reset.**
  - HS is the process that communicates with the watchdog timer.
  - HS monitors and reports on CPU usage
- **HS monitors:**
  - Critical applications defined in the Applications Monitor Table (AMT)
  - Critical events defined in the Event Monitor Table (EMT)
  - Message actions defined in the Message Actions Table (MAT)
  - Execution counters defined in the Execution Counter Table (XCT)
- **The HS Tables define actions taken by default.**
  - Example: Failure of a critical application results in
    - Processor Reset
    - Send Event, or
    - No Action
- **Tables can be replaced and reloaded by ground controllers**

HS Communications

- **All HS communications are Software Bus messages.**
  - Outgoing event messages are sent through calls to CFE_EVS_SendEvent. There are 70 HS events in Build 4.0
  - Incoming messages are received on 3 pipes (type CFE_SB_PipeId_t) :

CmdPipe	Ground Commands
EventPipe	Critical events generated by the 5 cFE applications
WakeupPipe	Wakeup message

- **HS Commands**
  - Prepare and send housekeeping data
  - No-op
  - Reset
  - Enable or disable application monitoring, event monitoring, aliveness
  - Reset the reset counter
  - Set the maximum number of resets

Formulate New Tests to Run in ITC

- **We tested HS in the ITC environment by:**
  - Triggering selected untested HS event messages.
  - Attempting to find a way to trigger a safe-hold state
  - Verifying that the wakeup pipe event works
    - It can never be triggered from the ground and is unused by any other application.
    - Wakeup may be left over from the legacy LRO/SDO missions.
- **Understand the HS application, requirements and tests.**
  - What are its major responsibilities?
  - How does it communicate with the operators and the other applications?
  - Was every requirement tested?
  - Did the software pass all of the tests?
- **Specifics considered for the new tests**
  - Look at every possible cause of a processor reset
  - Kill select application processes to verify that a processor reset is triggered and the applications are properly restarted
  - Fill the event pipe with events faster than they can be processed
  - Load badly formed tables and determine how the application reacts.

ITC HS Testing Results (Example 1)

- **Wake-up Message Pipe – Is it used or needed?**
  - HS listens for the wake-up message on the wakeup pipe (GPM Build 4.0).
  - There is no ground command to cause a HS wake-up.
  - There is no GPM HS requirement for a wake-up message
  - ITC testing required that a hexadecimal message be constructed and sent to trigger the wake-up since a ground command for it does not exist.
- **Sequence of Steps in ITC**
  1. Start up ASIST application and Simics application
  2. From the operator input console, start the HS application
  3. Use the debugger to set a breakpoint after the wakeup
  4. Watch for return status from the wake-up or timeout
     a) Status=0 (which is CFE_SUCCESS) indicated continue normal processing
     b) Manually enter the raw wake-up message at the console:

        ut_sendrawcmd "gc_hs", ("18B0C00000010000BC")

     d) Continue to watch the return value, with and without the raw command
     e) If the wake-up message was received, an early timeout occurred and normal processing continued
     f) Testing thus indicated that the code worked as intended

ITC HS Testing Results (Example 1 cont.)

- **Although the code works as intended:**
  - There is no GPM HS Level 5 Requirement for this functionality
  - An entire message pipe is dedicated to this single message
- **Submitting the issue in ORBIT**
  1. HS Wake Up Message ID processing is not needed since the message is never used. A message pipe is allocated for this one message.
  2. If the wakeup message is never received, this becomes a simple 1.2 second delay.
  3. GSFC-STD-1000E Section 3.02: "Elimination of Unnecessary and Unreachable Software"
     a) Justification is required for unused code.
     b) Focus must be on technical risk to the mission, not cost.
     c) Unnecessary code is generally not tested and could introduce a technical risk.
- **ORBIT State (8/8/2011): Submitted**
  - Comment from the GSFC POC:

    "The HS application reads from the pipe in line 249. Whether or not messages are sent to that pipe is mission-specific."

ITC HS Testing Results (Example 2)

- **Inconsistency between requirement and code for HS cFE power-on reset processing.**
  - Requirement HS8000 specifies counters to be initialized at power-on reset. Through code inspection, it appears that 11 of them are not reset, including:
    - Critical Application Monitoring status per Application Enabled
    - Watchdog Timer Flag set to TRUE
    - Peak CPU Utilization
    - Number of cFE Processor Resets
- **Sequence of Steps in ITC**
  1. Start up ASIST application and Simics application
  2. From the operator input console, start the HS application
  3. Use the debugger to set a breakpoint after the power-on reset processing
  4. Modify several of the variables to simulate a longer run
  5. Trigger a power-on reset from the command console
  6. Examine the value in each required flag or counter, and compare them with the requirements in HS8000 and the PLATFORM_DEFINED values in the code

ITC HS Testing Results (Example 2 cont.)

- **ITC testing revealed that certain values were not being reset.**
- **Examination of the GSFC test logs showed the same results**
- **Submitted ORBIT issue**
  - Impact: These parameters, if not reset could cause data processing problems with the cFE HS application.
  - Recommended action: A review of requirement HS8000 to determine if the additional reset parameters need to be accounted for within the HS code or if requirement HS8000 needed to be changed and the code to remain unchanged, based upon the needed functionality determined by the GPM Project.
  - Resolution Chronology: Lead Analyst discussed this with the GSFC POC. He accepted the issue and will present it to the GPM team.
- **ORBIT State (8/8/2011): To Be Verified**

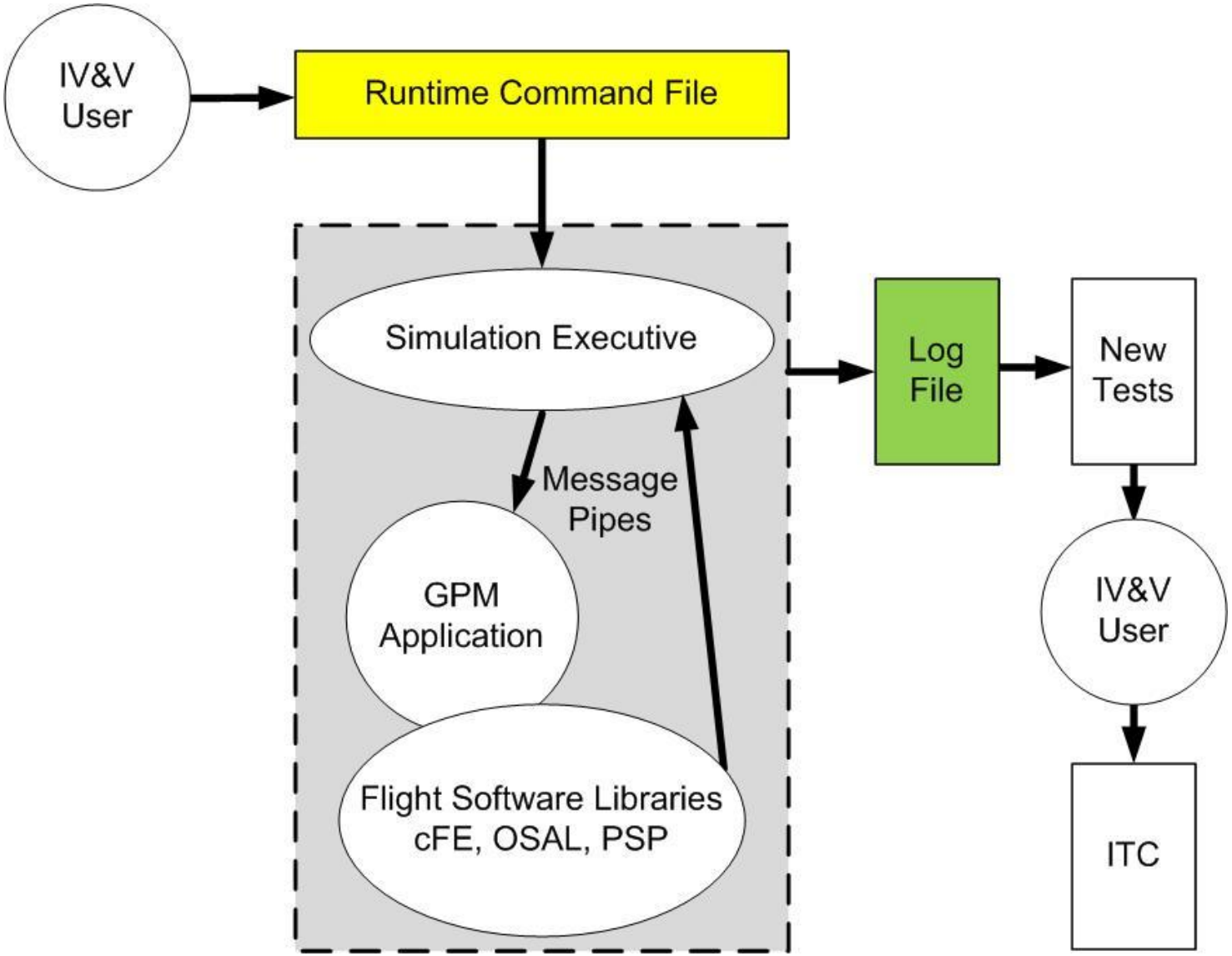
Lessons Learned from HS Testing in ITC

- **Although raw commands and code modifications can be sent to the application, these are not realistic in an operational scenario**
  - During GSFC code reviews, it has been specifically pointed out that the software is not designed to handle every possible raw command that could be sent during testing
- **GSFC tests the requirements, and repeating those tests is not productive.**
  - However, in some cases the GSFC tests do not appear to completely test each requirement. These are good candidates for testing.
- **Low-level scenarios can be tested. Examples in HS:**
  - Monitor resource margins such as CPU hogging and free memory space.
  - Monitor critical applications and events
  - Load new and/or corrupted HS tables
  - Triggering the use of an unneeded communications pipe
- **High-level scenarios cannot readily be tested.**
  - Some possible examples include:
    - Achieve orbit
    - Enter safe-hold
    - Exit safe-hold

Process Forward

- **How can we make more effective use of ITC?**
  - ITC testing might yield more ORBIT issues if applied earlier in a mission's software development process
  - The process for applying ITC is still evolving, even as the ITC itself evolves.
  - How can we formulate more complex system-level scenarios such as enter and leave safe-hold?
- **What if we could use a complementary dynamic test to identify possible new tests for the ITC?**
  - ITC provides a system-level test environment with multiple processes
    - Ideally it should simulate the spacecraft.
    - Some issues such as simulating the 1553 bus are still in development.
  - An application-level test would test the actual application in a simulated, single process environment - a software test-harness.
    - The test harness should be fast and simple
    - It would read test scripts that could be reused from build to build.
  - This could provide a dynamic way to identify tests for ITC, much like the current process for static code analysis.
  - A Simulation Executive could run on a PC without the expensive Simics licensed software

Test-harness Design Context



Conclusion

- **Most of the scenarios tested to date in the ITC demonstrated that the GPM software worked as expected.**
- **The GPM Team continues to refine a process for testing in the ITC environment.**
- **Improving the ability of Projects to develop effective test scenarios through a tool such as the Simulation Executive proposed in this Presentation will promote integration of dynamic testing within IV&V processes.**

Questions?

Phil Wheeler
philip.n.wheeler@ivv.nasa.gov
304-472-9520 x1211
Greg Black
gregory.j.black@ivv.nasa.gov
681-753-5253

Backup Slides

Review the COTS Lab Test Results

- **Test results of HS 2.1.0.0 were delivered with GPM Build 4.0.**
  - Builds 3.1 and 3.2 included HS 2.0.0.0
- **The following test scenarios were documented:**
  - Application monitoring (19 requirements)
  - Event monitoring (15 requirements)
  - Execution counter management (6 requirements)
  - General commanding (12 requirements)
  - Reset (13 requirements)
  - Watchdog management (5 requirements)
- **As of Build 4.0, all known HS requirements are implemented and documented as tested.**
- **Most requirements are tested at GSFC by demonstration. There are exceptions:**
  - About 1/3 of the possible events from HS were untested.
  - A few of the test scripts have typographical errors that prevent a small number of possible errors from being reported if they occur.
  - Test log files are missing for requirements HS6008, HS6009, HS6010, HS6011 and HS6012.